NASA's Instrument Control Markup Language (ICML)

Troy J. Ames, NASA/Goddard Space Flight Center
Kenneth B. Sall and Craig E. Warsaw, Century Computing, Inc.

Keywords:
astronomy: infrared, infrared astronomy, XML, instrument: description, telescope: control, SOFIA, NASA: GSFC, software: methodology

Abstract

This paper describes a real-world experience in using XML (Extensible Markup Language) to aid in collaborative scientific research involving infrared astronomy. We describe our current efforts for the Stratospheric Observatory For Infrared Astronomy (SOFIA) in developing an XML vocabulary for instrument description, communication and control, pipeline algorithms, metadata, and documentation. The initial SOFIA instruments to be controlled are the High-resolution Airborne Wideband Camera (HAWC) and ultimately the Submillimeter And Far InfraRed Experiment (SAFIRE). The focus of this joint effort between NASA/GSFC's Advanced Architectures and Automation branch (Code 588) and Century Computing has been infrared astronomy, although most of the techniques employed have much wider applicability. Our vocabulary, originally called ICML, has been recently renamed \em AIML (Astronomical Instrument Markup Language).

Instrument Remote Control

Instrument Remote Control (IRC) is a joint effort between NASA/GSFC's Advanced Architectures and Automation branch (Code 588) and Century Computing. IRC will enable trusted infrared astronomers around the world to easily access infrared astronomical instruments located in remote, inhospitable environments.

Project Goals and Methodology

The long-term goal of the Instrument Remote Control (IRC) project is to develop a distributed framework from science user to instrument which will provide robust interactive and reconfigurable control and monitoring of remote instrumentation. It is crucial that the next generation of astronomical instruments be largely free of the constraints imposed by specific operating systems and rigid computer hardware configurations that are likely to need to be changed many times during the lifetime of the instruments. IRC promotes the idea of distributed environment with processes assigned to the CPU and hardware best suited to the task. Our assumption is that better equipment will appear soon after commissioning; we must therefore be able to accommodate changes easily. Our software methodology is characterized by several key goals and principles:

- It must be relatively easy to develop, enhance, and maintain the software.
- Addition of a new instrument should be an incremental, rather than monumental, task.
- Object-oriented design techniques facilitate the development of a general solution that can be applied to a diverse range of instruments.
- The methodology must support instrument integration, testing and control.
- IRC explicitly promotes reuse -- it is part of our design, not an afterthought:
  - Reuse applies to different instruments as well as to reuse of sub-components.
  - Various observatories can reuse portions of code and instrument description.
  - We will incorporate emerging technologies that promote reuse.

To accomplish our goals, we are developing a comprehensive instrument description based on Extensible Markup Language (XML)[1], a World Wide Web Consortium standard. This description will drive nearly all of our software development efforts. We will make use of automated tools to assist in the definition of this description, as well as in incorporating changes to the description. Our ultimate system will interpret the descriptions at runtime, where performance allows. In other cases, we will generate code from the latest instrument description.

Interested Parties

Our initial prototype was a Java application developed for the Center for Astrophysical Research in Antarctica (CARA) that ran on a Mac Quadra to control heaters for SPIREX at the South Pole in early 1998. IRC is currently focusing on the design and implementation of the control software for HAWC, the High-resolution Airborne Wideband Camera, for SOFIA (Stratospheric Observatory For Infrared Astronomy, http://sofia.arc.nasa.gov/). Our second SOFIA instrument will be the Submillimeter And Far InfraRed Experiment (SAFIRE). Both HAWC and SAFIRE are SOFIA First Light Instruments (http://www.sofia.usra.edu/observatory/instruments/first_light/announce.html). We believe our techniques will also be applicable to ESA's SPIRE (Spectral and Photometric Imaging REceiver for FIRST - Far InfraRed and Submillimetre Telescope) as well. These three instruments have some features in common (e.g., their detectors) which makes them good candidates. However, we emphasize that our techniques can be extended to any astronomical instruments. In fact, our solution can only become truly general if we have a diverse set of data points; we encourage members of the astronomical community who share our concerns to contact the authors to contribute in some manner to this development effort.

Astronomical Instrument Markup Language (AIML)

Our long-term focus is to develop an extensible framework to which new instruments can be added with relative ease. This will eventually be accomplished by implementing our own *Astronomical Instrument Markup Language (AIML)*[2] based on a custom Document Type Definition (DTD) which will evolve over time, as a direct effort of NASA/GSFC and Century Computing. Since AIML is an XML vocabulary, it will be well suited to describing hierarchical, structured information. AIML will be used to describe control capabilities, data streams, message formats, and communication mechanisms, as well as for online documentation and the association of housekeeping metadata with acquired images. Some of these aspects of instrument control will be reflected in Java graphical user interfaces, generated from the instrument descriptions. Other sections of the instrument description will be applied to data capture and data pipeline algorithms, as well as to other instrument subsystems. Different parts of the description can be used by different components of the software architecture, and even by separate applications. Furthermore, once we collect a number of instrument descriptions from a variety of observatories, astronomers will be able to specify what characteristics they need for a particular proposal and generate a query that will return a list of suitable instruments.

Applicability of XML to IRC

There are many aspects of instrument description, algorithm description, and other documentation that are well suited to the hierarchical, structured markup that XML facilitates.

- Control Capabilities - what the instrument can do
- Data Stream Descriptions

[1] Several key XML URLs appear in the References.
[2] The original name for our language was *Instrument Control Markup Language (ICML).* However, since control is no longer our only focus, we decided to rename our language *Astronomical Instrument Markup Language (AIML).*

- images - data collection, archiving, indexing, and retrieval
- housekeeping - instrument status, health and safety
- command responses

♦ Pipeline algorithms - inputs and outputs of algorithms; connecting algorithms; defining visualizations as consumers of the pipeline outputs
♦ Message Formats - separate formats for Control and Data Streams
♦ Communication Mechanisms - ports, TCP, RS232
♦ Documentation
- template with specific styles applied
- novice vs. expert views of documentation
- role-dependent view - investigator, operator, or astronomer role
- short descriptions - displayed as tool tips in the GUI
- long descriptions - multiple sentences or paragraphs for on-line help

♦ Style and Presentation
- XSL (Extensible Style Language) for GUI components and eventually layout
- if the XML description contains a range of values, generate a slider (scale) in the GUI
- if the XML description contains enumerated choices, generate a list or radio buttons

♦ Metadata / Housekeeping

Down the road, we expect to expand AIML to encompass:
♦ Planning Observations (proposals)
♦ Scheduling Instruments for Observations - hooks to read instrument description, timing constraints, hardware limitations, etc.

The examples in the following subsections are meant to be illustrative of our approach.

Metadata Example

The AIML example below is a simple description of an element we call an
\em Image\em which consists of two parts: (1) the data itself, stored in a separate file;
and (2) the metadata which defines the circumstances under which the data was collected.
Although the example specifies FITS, our notion of an Image is independent of any particular
binary representation. A key advantage of this approach is that the metadata portion can
be processed by any XML-aware tool and is fully searchable, regardless of the binary format
which the metadata describes.

```
<Image>
      <Data href="m42th1a.fits" type="FITS" series="1247"/>
      <Metadata>
        <ObjectName>M42, Orion Nebula, Theta-1</ObjectName>
        <TimeStamp>
            <Date month="09" date="16" year="2001" />
            <Time hrs="17" min="24" sec="13" msec="145" />
        </TimeStamp>
        <Integration duration="20" filter="300micron" />
        <FlightInfo ra="5:33" dec="5:25" />
                <Notebook>This stellar image was taken on our maiden
voyage. Weather conditions were...
        </Notebook>
      </Metadata>
</Image>
```

We recently learned of Astronomical Markup Language (AML,
http://www.infm.ulst.ac.uk/%7Edamien/these/) by Damien Guillaume and Fionn Murtagh. AML describes

astronomical objects, articles, tables, set of tables, images, and authors. We hope to coordinate our efforts with those of Guillaume and Murtagh. Note that our AIML description is aimed at the wider domain of instrument description and control, as well as data collection and pipeline algorithms.

ADR Subsystem Example

The next example describes two Commands, `HeatSwitch` and `HeatSwitchCurrent`, both of which accept exactly one argument. Note that the "current" argument of the `HeatSwitchCurrent` command has `Valids` (constraints) defined. This results in a GUI component that will guarantee input only within the constrained range.

```
<Control id="HAWC.ADR">
  <Command id="HeatSwitch">
    <Argument id="state" type="boolean" required="true"/>
    <!-- other Arguments would go here -->
  </Command>
  <Command id="HeatSwitchCurrent">
    <Argument id="current" type="float" required="true">
      <Valids>
        <Range low="0.0" high="200.0" units="amps" />
      </Valids>
    </Argument>
  </Command>
  <!-- other Commands would ge here -->
</Control>
```

Summary

We have attempted to convey the significant advantages that would result from describing astronomical instruments and data pipeline algorithms using AIML, our XML vocabulary. Separation of GUI descriptions from application logic has long been a favored practice in the software industry. It is time for the astronomical community to favor an approach that similarly separates out as much of the instrument details from the code as possible. By so doing, we can develop instruments which can be developed and modified far more quickly than those created by conventional software development techniques. We will also produce a database of instrument descriptions that can greatly simplify the task of locating the right instrument for achieving specific proposal goals. Join our low-volume XML for astronomy mailing list by sending a one-line message to majordomo@pioneer.gsfc.nasa.gov :

```
subscribe xml your_email_address
```

References

NASA/Goddard Space Flight Center 1998, SOFIA HAWC Public Web site, http://aaaprod.gsfc.nasa.gov/hawc/
NASA/Ames Research Center 1998, SOFIA Home Page, http://sofia.arc.nasa.gov/
Web Developer's Virtual Library: XML home page, http://WDVL.com/Authoring/Languages/XML/
World Wide Web Consortium 1998, XML Activity Statement, http://www.w3.org/XML/Activity.html